04

OpenCV 사용하기 II

HSV 활용하기

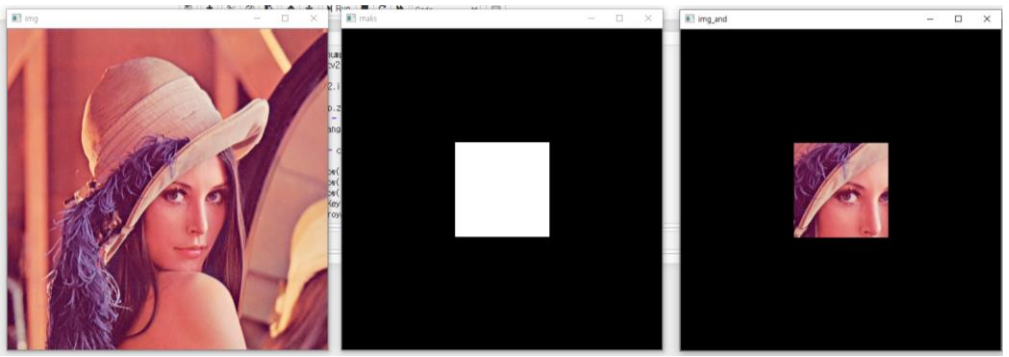
04 cv2.bitwise_and

- **학습 목표:** 0, 1을 가지고 하는 연산으로 두 이미지의 동일한 위치에 대한 연산을 진행.

- **메인 코드:**

```python
import numpy as np
import cv2

img = cv2.imread("Image/Lenna.png", cv2.IMREAD_COLOR)

mask = np.zeros(img.shape[:2], dtype = "uint8")
(cX, cY) = (img.shape[1] // 2, img.shape[0] // 2)
cv2.rectangle(mask, (cX - 75, cY - 75), (cX + 75, cY + 75), 255, -1)

img_and = cv2.bitwise_and(img, img, mask=mask)

cv2.imshow("img", img)
cv2.imshow("maks", mask)
cv2.imshow("img_and", img_and)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

- **결과:**



04 cv2.bitwise_and

- **코드 설명:**

1. cv2.bitwise_and(이미지1, 이미지2, 영역)

   비트연산은 이미지에서 특정 영역을 추출할 때 유용하게 사용된다. 예를 들면 이미지에서 바탕을 제거하고, 2개의 이미지를 합치는 경우입니다.

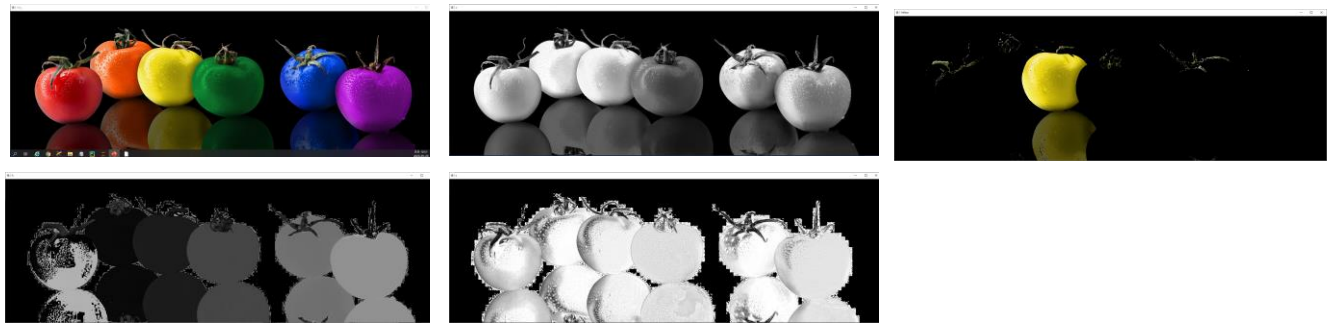
04 hsv

- **학습 목표:** Hue(색상), Saturation(채도), Value(명도)로 분리하여 이미지를 검출하기 위해 사용합니다.

- **메인 코드:**

```
In [*]: import cv2

img = cv2.imread("Image/tomato.jpg", cv2.IMREAD_COLOR)
hsv = cv2.cvtColor(src, cv2.COLOR_BGR2HSV)
h, s, v = cv2.split(hsv)

y_h = cv2.inRange(h, 25, 35)
yellow = cv2.bitwise_and(hsv, hsv, mask = y_h)
yellow = cv2.cvtColor(yellow, cv2.COLOR_HSV2BGR)

cv2.imshow("Yellow", yellow)
cv2.imshow("img", img)
cv2.imshow("h", h)
cv2.imshow("s", s)
cv2.imshow("v", v)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

- **결과:**



04 hsv

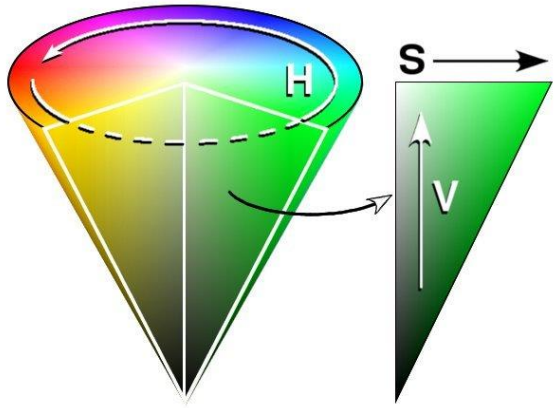
- **코드 설명:**

1. cv2.cvtColor(원본이미지, 색상변환코드)

   이미지의 색상을 변경할 수 있습니다.

   색상 변환 코드는 원본 색상 코드 2 결과 색상 코드로 이루어집니다.

   ex) BGR2HSV = BRG (RGB 역순) 2 HSV



2. cv2.split()

   각 속성별로 채널을 분리합니다.

3. cv2.inRange(단일 채널, 최소값, 최대값)

   단일채널: Hue(색상)의 범위를 조절하여 특정 색상만 표시합니다.

04 Split & Merge

- **학습 목표:** 이미지의 채널을 분리하고 병합하기 위해 사용합니다.
- **메인 코드:**

```python
import cv2

img = cv2.imread("Image/tomato.jpg", cv2.IMREAD_COLOR)
b, g, r = cv2.split(img)

cv2.imshow("r", r)
cv2.imshow("g", g)
cv2.imshow("b", b)

meg = cv2.merge((b, g, r))

cv2.imshow("merge img", meg)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

- **결과:**

04 Split & Merge

- **코드 설명:**

1. cv2.merge ((채널1, 채널2, 채널3))

   나누어진 채널을 다시 병합할 수 있습니다.

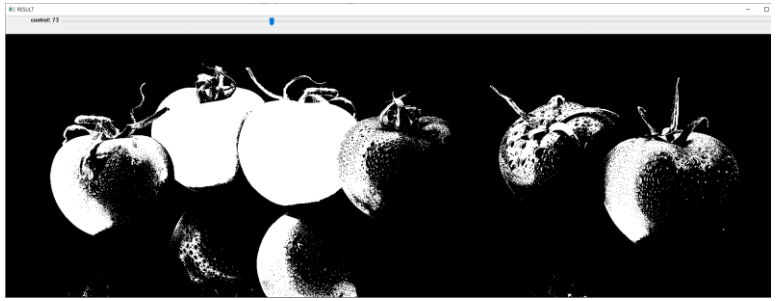
04 Trackbar

- **학습 목표:** 트랙바를 생성하여 일정 범위내의 값을 변경합니다.
- **메인 코드:**

In [1]:
```python
import cv2

def nothing(x):
    pass

img = cv2.imread("Image/tomato.jpg")
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

cv2.namedWindow('RESULT')
cv2.createTrackbar('control', 'RESULT', 0, 255, nothing)
cv2.setTrackbarPos('control', 'RESULT', 0)

while True:
    value = cv2.getTrackbarPos('control', 'RESULT')

    ret, thre = cv2.threshold(gray, value, 255, cv2.THRESH_BINARY)
    cv2.imshow("RESULT", thre)

    if cv2.waitKey(1) & 0xff == ord('q'):
        break

cv2.destroyAllWindows()
```

04 Trackbar

- **결과:**



- **코드 설명:**

1. cv2.createTrackbar (트랙바 이름, 윈도우창 이름, 최소값, 최대값, 콜백 함수)

   트랙바를 생성합니다. 트랙바의 최소, 최대값을 지정하고 콜백함수를 통해 바를 조절시 값을 반환합니다.

2. cv2.setTrackbarPos (트랙바 이름, 윈도우창 이름, 설정값)

   트랙바의 값을 설정하며, 설정값에는 초기값을 할당할 때 사용합니다.

3. cv2.getTrackbarPos (트랙바 이름, 윈도우창 이름)

   트랙바의 현재 위치를 리턴하기 위해 사용

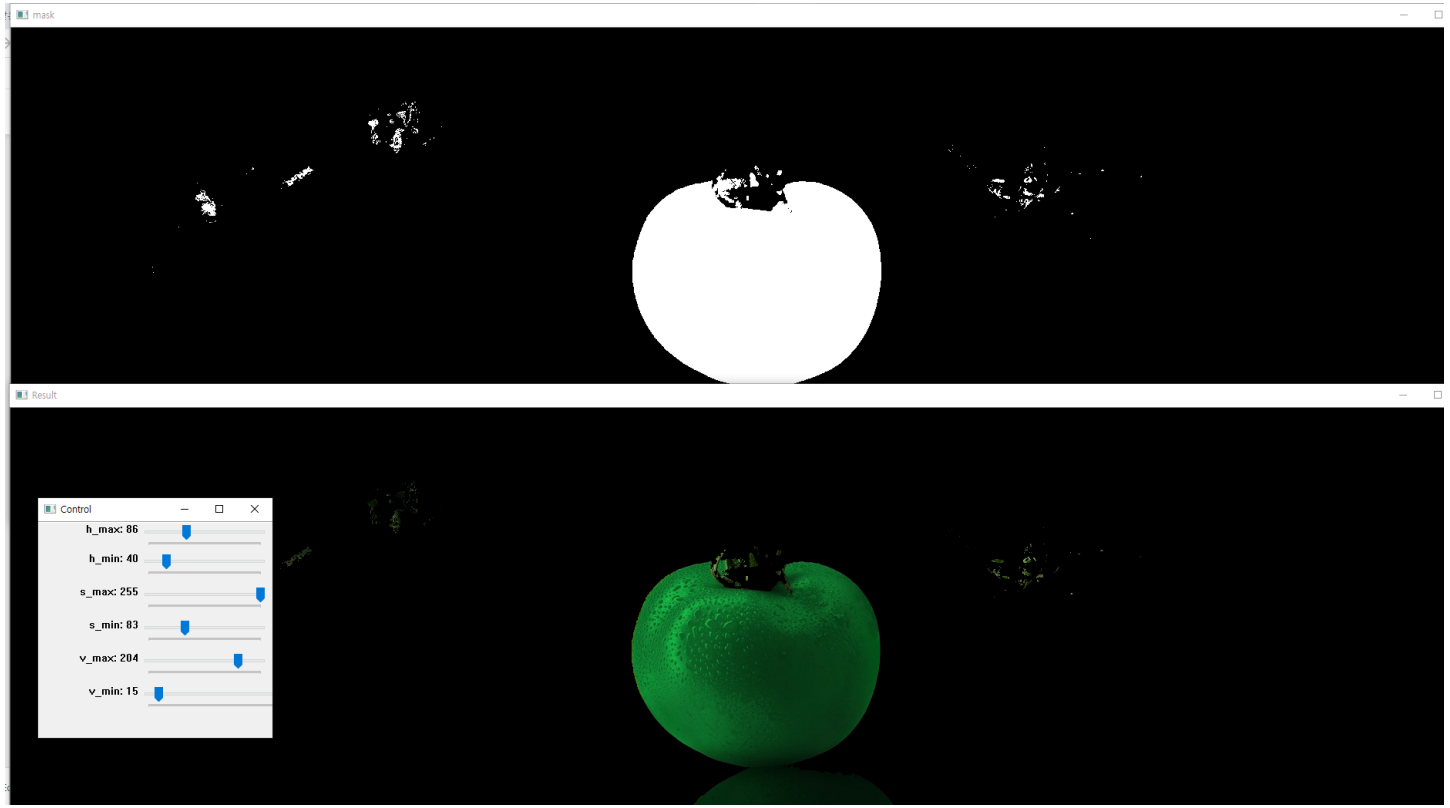
04 Hsv process

- **학습 목표:** 트랙바를 사용하여 HSV 값 조절하기.
- **메인 코드:**

In [1]:
```python
import cv2

def nothing(x):
    pass
def create_control_window():
    cv2.namedWindow('Control')
    cv2.createTrackbar('h_max', 'Control', 0, 255, nothing)
    cv2.createTrackbar('h_min', 'Control', 0, 255, nothing)
    cv2.createTrackbar('s_max', 'Control', 0, 255, nothing)
    cv2.createTrackbar('s_min', 'Control', 0, 255, nothing)
    cv2.createTrackbar('v_max', 'Control', 0, 255, nothing)
    cv2.createTrackbar('v_min', 'Control', 0, 255, nothing)

    cv2.setTrackbarPos('h_max', 'Control', 255)
    cv2.setTrackbarPos('h_min', 'Control', 0)
    cv2.setTrackbarPos('s_max', 'Control', 255)
    cv2.setTrackbarPos('s_min', 'Control', 0)
    cv2.setTrackbarPos('v_max', 'Control', 255)

def get_control_value():
    h_min = cv2.getTrackbarPos('h_min', 'Control')
    h_max = cv2.getTrackbarPos('h_max', 'Control')
    s_min = cv2.getTrackbarPos('s_min', 'Control')
    s_max = cv2.getTrackbarPos('s_max', 'Control')
    v_min = cv2.getTrackbarPos('v_min', 'Control')
    v_max = cv2.getTrackbarPos('v_max', 'Control')

    high_val = (h_max, s_max, v_max)
    low_val = (h_min, s_min, v_min)

    return high_val, low_val

img = cv2.imread("Image/tomato.jpg")
hsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)

create_control_window()

while True:
    high_val, low_val = get_control_value()
    mask = cv2.inRange(hsv, low_val, high_val)
    cv2.imshow('mask', mask)

    result = cv2.bitwise_and(hsv, hsv, mask=mask)
    result = cv2.cvtColor(result, cv2.COLOR_HSV2BGR)

    cv2.imshow("Result", result)
    if cv2.waitKey(1) & 0xff == ord('q'):
        break

cv2.destroyAllWindows()
```

04 Hsv process

- **결과:**



04 문제 3. 색상별로 이미지 추출하기

- **문제:**

  HSV이나 RGB 중 한 개의 색영역을 이용하여 색상별로 토마토 사진을 추출합니다.

  사진에 포함된 토마토는 다음과 같습니다.

  빨간 토마토　　주황 토마토　　노란 토마토

  초록 토마토　　파란 토마토　　보라 토마토

  *HSV에서 빨간색은 처음과 끝값에 위치하므로 완전히 나오지 않아도 괜찮습니다.

- **결과:**